**Personality Identification With Myer Briggs**

*A project report submitted in fulfillment of the requirement for the*

BAN 675 – Text Mining

Submitted by

Xiaodong Nie – qs6942

Dorbens Jean-Pierre - cp4897

Tina Hu – if2392

Soon Chye Lim – jn7736

Glee Truong – gq3622

Master of Science in Business Analytics

California State University, East Bay

Under the guidance of

**Dr. Peng Xie**

****

**COLLEGE OF BUSINESS AND ECONOMICS**

**CALIFORNIA STATE UNIVERSITY, EAST BAY**

**TABLE OF CONTENT  
  
  
  
1. Abstract**[**:**](https://docs.google.com/document/d/1s8YpWCONEOAja5qnwOCKiMsnj3facpyD0FP2GG2Bux8/edit#heading=h.6sovgoweqdmn)3

**2. Introduction:** 3

2.1 Myer Briggs 16 Personality Types [3](https://docs.google.com/document/d/1s8YpWCONEOAja5qnwOCKiMsnj3facpyD0FP2GG2Bux8/edit#heading=h.amwuttolw80z)

2.2 Data Description 3

**3**. [**Data Exploration and Preparation:**](https://docs.google.com/document/d/1s8YpWCONEOAja5qnwOCKiMsnj3facpyD0FP2GG2Bux8/edit#heading=h.uxexeuyvb0j0)[3](https://docs.google.com/document/d/1s8YpWCONEOAja5qnwOCKiMsnj3facpyD0FP2GG2Bux8/edit#heading=h.uxexeuyvb0j0)

3.1 Preprocessing 4

3.2 Cloud Word 4

3.3 Lexical Diversity 6

3.[4](https://docs.google.com/document/d/1s8YpWCONEOAja5qnwOCKiMsnj3facpyD0FP2GG2Bux8/edit#heading=h.hnghd7hxphyz) Emoticon/Emoji Conversion 6

3.5 Features/One Hot/Labels 7

**4. Methodologies:** 8

4.1 Deep Learning 8

4.2 Decision Tree 10

4.3 Support Vector Machine 11

**5. Conclusion:** 14

**6. Appendix:** 15

**1. Abstract**

Our main goal is to predict the individual’s Myers-Briggs personality based on their online post. We ask ourselves a few research questions before we start constructing our models. The first question is what features in the post would help predict the personality. Second, which model will give us the best accuracy? And finally, would the model be valuable and feasible for the public to use after determining the best model? These questions are to be answered later in the sections.

**2. Introduction**

As humans we are very intrigued in finding out why or being able to explain why we do things. Take for example Zodiac signs and use of Astrology. These two are used widely in Millenials and Gen Z to see if they are compatible with people or what their future will look like. Unlike these, the Myer-Briggs Type Indicator (MBTI) is based on how you answer questions. MBTI is a self-report questionnaire that attempts to understand who you are psychologically by the way you answer questions about the world. This test can be rigged if the individual wants to light themselves in a certain way and answer the questions how they’d like others to perceive them.After the individuals truthfully finish the questionnaire it’ll then classify them into 1 out 16 personality types.

**2.1 Myer Briggs 16 personality Types**

The MBTI has 16 possible personalities. They are a mixture of types from 4 possible categories: 1) (**I**)ntrovert or (**E**)xtrovert, 2) (**S**)ensing or I(**N**)tuition, 3) (**T**)hinking or (**F**)eeling, 4) (**J**)udging or (**P**)erceiving.The test compiles a type from each category based on the user's response to certain questions. For example an INFJ is someone who is idealistic,compassionate,organized and enjoys intellectual pursuits.

**2.2 Data Description**

For this project we used a dataset from Kaggle. The dataset records 8675 individual accounts and what they have posted on the particular forum. Each individual knows their personality from taking the MBTI test and that is tied to their post.



*Figure 1 Dataset Example*

**3. Data Exploration**

**3.1 Preprocessing**

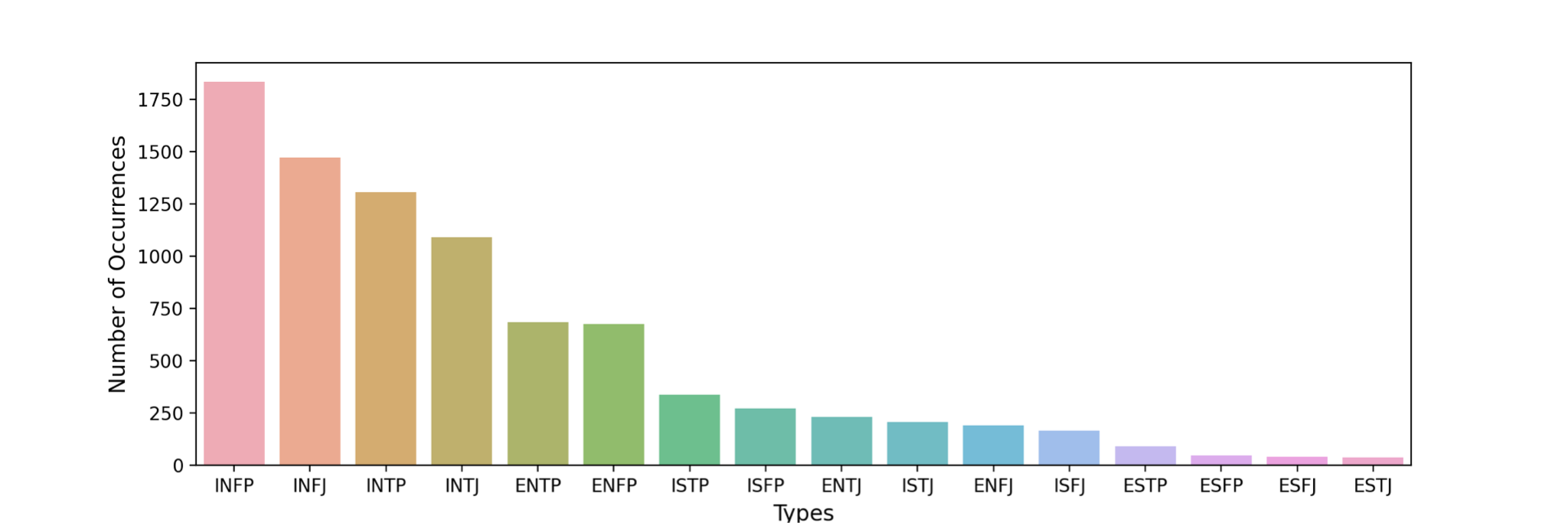
As you can see from the picture of the first 10 individuals, the posts need to be cleaned. The first thing we did is remove urls, delimiters (|||), and numbers. Then we tokenized each user's post and then lowercased them so we’d be able to remove the English stopwords. After, we Lemmatized the list of words and removed the punctuations which gave us our final tokenized list of words.



*Figure 2 Cleaned Posts*

**3.2 Cloud Word**

After we examined our dataset, we were surprised to see the unbalanced personality types in our dataset. (see fig.1). The top type is INFP with more than 1750 posts; the ESTJ, the minor group, only has less than 50 posts. To solve the skewness, we separated each MBTI type into four different groups. They are Extraversion (E) or Introversion (I), Intuitive (N) or Sensing (S), Thinking (T) or Feeling (F), and finally Judging (J) or Perceiving (P). Meanwhile, we will use these four groups as our labels of the classifier models.



*Figure 3 Personality Types of Distribution*

We assumed each letter type is independent of other types for the research work on the MBTI types. For example, the letter Introversion/Extraversion will not affect Thinking/Feeling.

We tried other solutions to explore the differences among these types, like two letters as a type, three letters as a type, and heatmap to explore the relationship between one and others. However, we believe the Word Cloud could give us the most effective visualization on this dataset. We generated 16 personality letters in Word Cloud. These figures were created by extracting each adjective from 16 groups of users. The size of each word indicates the proportionality of its appearance frequency used by the users. We considered this World Cloud would tell some exciting information that different types tend to use other words.

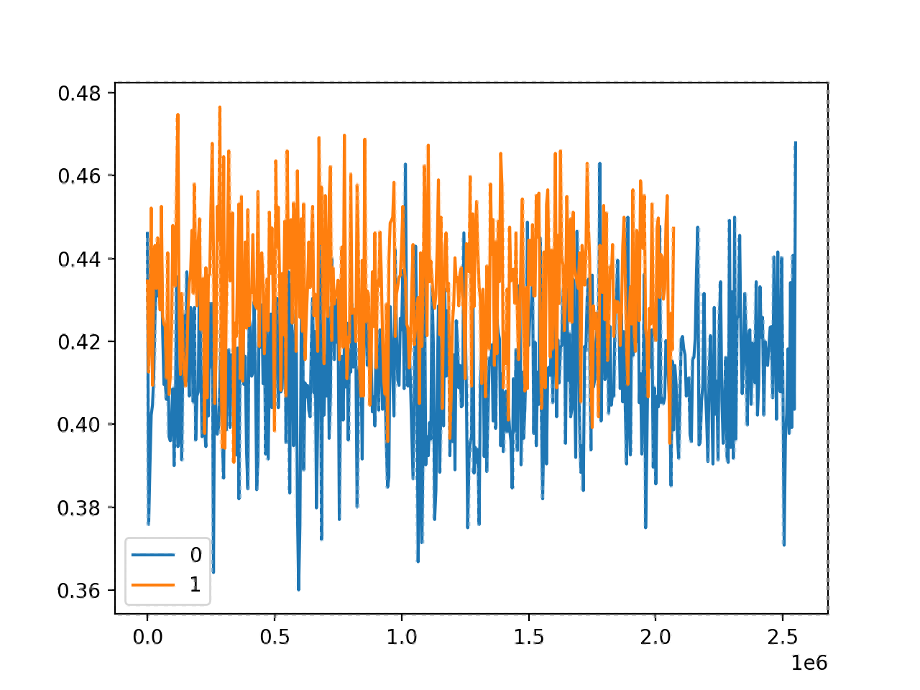


*Figure 4 Cloud Word (I)ntroversion (Left) vs. (E)xtraversion (Right)*

(fig. 2) gives us a hint that most of the words' usages are similar by comparing Introversion to Extraversion. The reason is that we all exhibit introversion and extraversion to some degree, but most of us will have an overall preference for one or the other. However, there are still some differences while investigating it. For instance, the words "many" and "little" appeared more in Introvert users than in Extravert users. Same as other type comparisons. (see Appendix IV)

**3.3 Lexical Diversity**

Lexical diversity (LD) is an important indicator to measure the complexity and difficulty of reading the text. We tended to figure out the complexity in each user type on their letters. In (fig.3) we can see the LD result between (T)hinking to (F)eeling. We can also see there is an apparent difference between this group. The number 1 or orange line indicates the T group, and the group has higher complexity text than the F group. It made sense because, from the MBTI, the thinkers tend to make decisions using logical analysis while feelers are more sensitive and cooperative.



*Figure 5 Lexical Diversity Between Thinking (T) to Feeling (F)*

In addition, from other groups’ LD outcomes, most of them have approximately the same difficulty index as T and F (0.42 - 0.44). Unlike groups N and S., their complex indicators are lower, around 0.4 - 0.25 (see Appendix V).

**3.4 Emoticons/Emoji conversion**

(fig.4) below shows the five most-used emojis for each personality. As you can tell, happy and laughing emojis were used most often among most users, except for ESTP and ESFJ people, who use more frown emojis than laughing emojis.

*Figure 6 Five Common-Used Emojis*

**3.5 Features/OneHot**/**Labels**

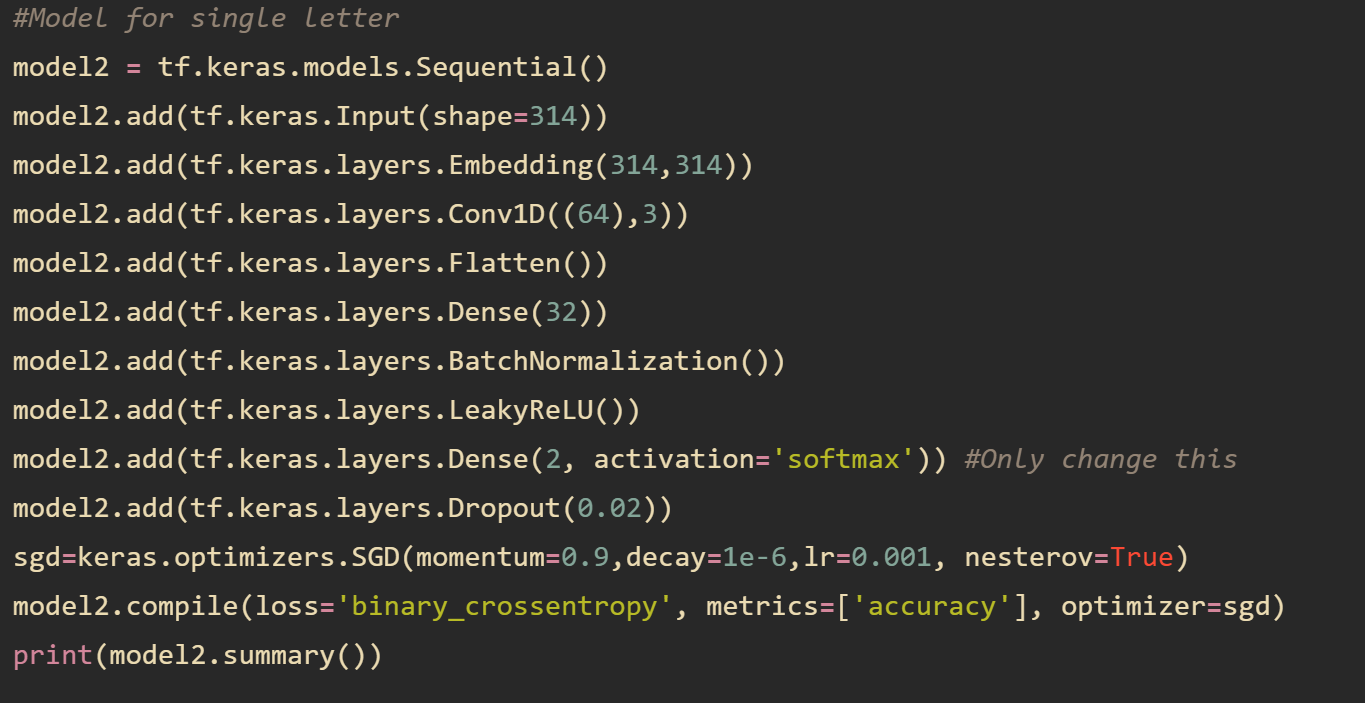
Looking at (fig.7) below represents our final dataset that we would be using in our models. Starting with the myerTypes, these are the user's self identified personality types. Then we have the users average sentence length per post, the average character per post, average TFIDF, Vader Score. From there we gathered 37 of the most used emoji, 45 of the most used adjectives, and 228 of the most used words out of the whole dataset. For our models to understand these words, we covered the individual emoji, adjectives, and words into one-hot encoding or dummies. This allowed us to feed information into the models reducing load times and error types. Lastly this leaves us with the 4 labels we would be using to compare our validation later on. With the labels we decided to keep the original users Myer personality and broke their personality into the four categories of : I/E, N/S, P/J, and F/T. To avoid redundancy, each personality is paired with their particular section, which allows us to create these labels that are similar to the dummies variables: P(1)/J, N(1)/S, I(1)/E, F(1)/T. Once the data was processed we split the data into a 80/20 training and testing dataset. The training dataset is used to fit the model, the test dataset is used to evaluate the fit model. This then allowed our team to choose the appropriate features and adjust the types of features according to the characteristics of the model.



*Figure 7 DataFrame in One Hot Encoding*

**4. Methodologies**  
**4.1 Deep Learning**

We normalized the data frame and changed the label data frame to one-hot encoding. These are required so the data will fit our neural network model. We can start constructing the deep learning model as shown below:

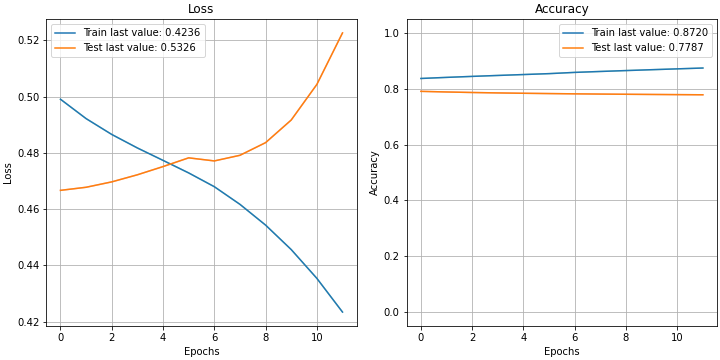


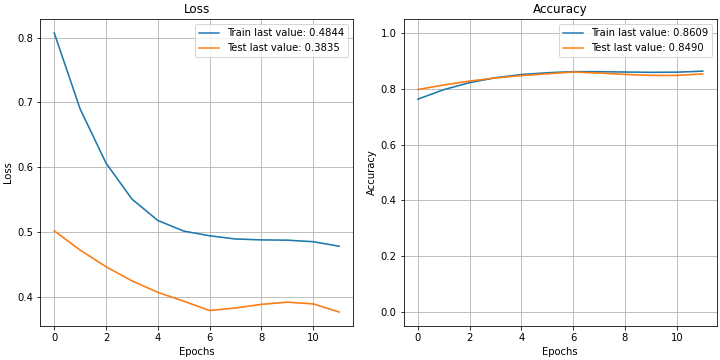
*Figure 8 (more examples in the appendix II)*

We introduced a variety of Keras layers in our sequential model. First, we defined our input shape to 314, which is similar to the number of variables in our data frame. The first layer is an embedding layer which is a prerequisite to our next layer, the convolutional layer. After feeding the data through the convolutional layer, it is flattened and fed into the dense layer. A batch normalization layer is added after the dense layer to speed up the process. Finally, our output layer is set to 2 neurons because of the binary classification (eg. Introvert versus Extrovert). Leaky Rectified Linear Unit (Leaky ReLu) is the activation function that allows small negative values if the input is less than zero. In total, there are about 800,000 trainable parameters in both models.

We implemented several regularization techniques in our model to prevent the model from overfitting. First, we added a dropout layer at the end, with 2% of the outputs being ignored. And setting our learning rate to 0.001 to ensure our loss curve achieves local/global minima and not oscillating.   
 Lastly, we compiled our model. Our loss model is binary cross-entropy, accuracy metrics, and sgd optimizer. We then split the data for 80% training and 20% validation. Finally, We ran the model with a batch size of 64 for 12 epochs. And below diagram is the result.

Introvert Versus Extrovert (I or E)



Intuition Versus Sensing (N or S)  


*Figure 9 (more examples in the appendix II)*

It is evident that most models are learning because the slope of the loss model goes down and then flattens out. Except for the I or E model, because it is overtrained and the generalization error is increasing after epoch 4. Therefore, we can just draw a line at epoch 4, and that will be our accuracy metrics for the IE model. Our training and valid accuracy were close to each other, so that means we have little generalization error. We will take the validation accuracy as our ultimate measure for the model’s performance. The below table shows the accuracy for all models.

| Model | Training Accuracy | Valid Accuracy | Accuracy Benchmark | Above Benchmark |
| --- | --- | --- | --- | --- |
| I or E | 0.8720 | 0.7787 | 0.5000 | Yes |
| N or S | 0.8609 | 0.8490 | 0.5000 | Yes |
| T or F | 0.8072 | 0.7533 | 0.5000 | Yes |
| P or J | 0.7351 | 0.6732 | 0.5000 | Yes |

*Table 1: Deep Learning Accuracy Table*

From the above results, we know that the model is best at predicting N/S followed by I/E, T/F, P/J, and Myer Briggs. All of them surpassed the naive benchmark calculated by 1 divided by the number of classes. Therefore, the neural network model can be one of the potential models for our analysis.

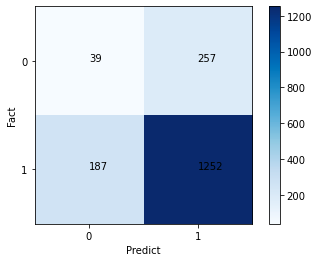
**4.2 Decision Tree**

The decision tree is a predictive model. Our decision tree is used to classify the class of the online post. To prevent overfitting, we choose four features to build the model. They are average sentence length, average word length, average TFIDF, and Vader sentiment score. We will use the validation accuracy, precision, recall, and F1 score to measure the performance of the five models. The below table shows the details.

| Model | Accuracy | Precision | Recall | F1 |
| --- | --- | --- | --- | --- |
| I/E | 0.6484 | 0.7670 | 0.7658 | 0.7664 |
| N/S | 0.7504 | 0.8730 | 0.8343 | 0.8532 |
| P/J | 0.5077 | 0.5986 | 0.5850 | 0.5917 |
| F/T | 0.5342 | 0.5681 | 0.5620 | 0.5651 |

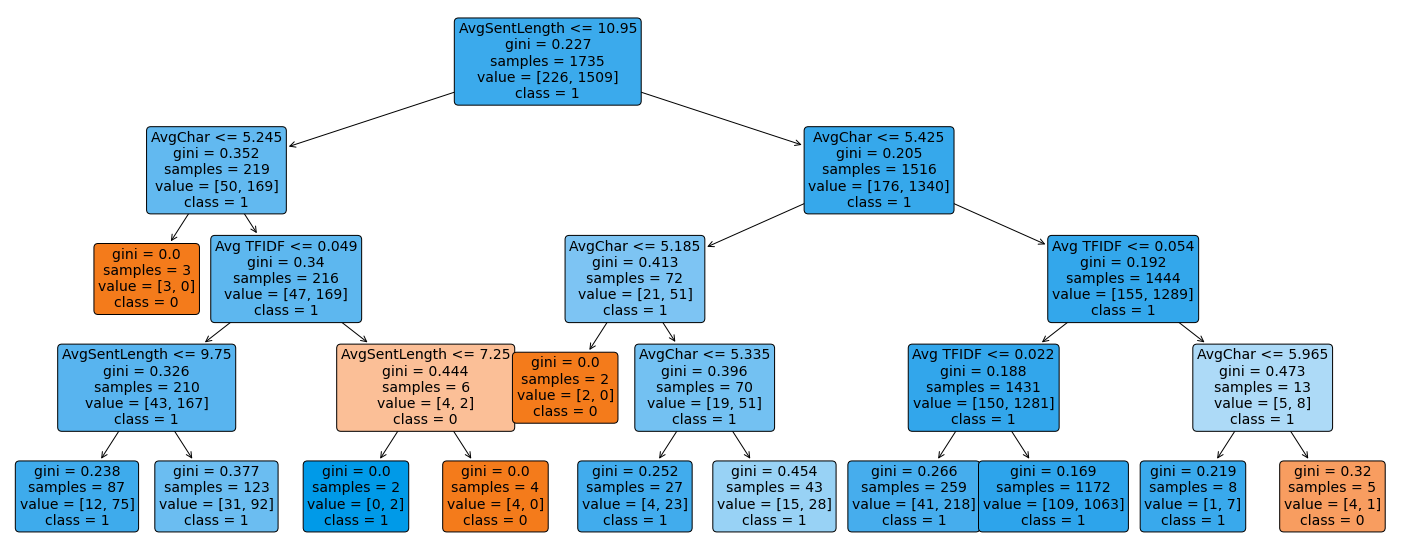
*Table 2 Decision Tree Model*

From the table we know, in these four models, the decision tree model is best to predict the N/S model. N/S has the highest accuracy of around 75%. The 87% precision shows a high true positive rate in total predicted positive observation. Next, we will use the confusion matrix and decision tree to analyze the N/S model in detail.



*Figure 10 N/S Model*

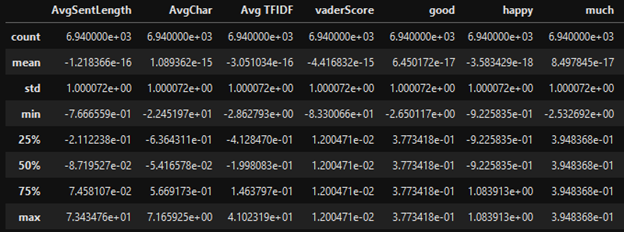
The confusion matrix (fig.6) shows valuable classification statistics. We mainly analyze the confusion matrix of the intuitive and sensing(N/S) model. The total actual intuitive personality class is the sum of the value of fact 1, the value is 1439. The total actual sensing personality class is the sum of the value on the fact 0, the value is 296. The value of true positive prediction is 1252, which means there is a 1252 correct prediction of intuition personality class. Also, 39 true negative predictions, shows the correct prediction of sensing personality class is 39 times. The confusion matrix clearly shows how to calculate from scratch and interpret the results.



*Figure 11 Decision Tree of N/S Model*

Decision trees are extremely easy to understand. Each feature choice and resulting outcome flow into each other. The graph above is the decision tree of the intuitive and sensing(N/S) model. The depth of this decision tree is 4, the blue block represents class 1(intuitive personality) and the orange block represents class 0 (sensing personality). We use the Gini criterion to create the decision tree. The root node is blue which represents the class 1 records more than class 0 records in the samples. According to the rule Gini, each node classifies a different feature and then belongs to different classes. When we check the terminal node, most of the branches belong to class 1. So the decision tree analysis allows better preparation for each likelihood and makes the most informed choices for each step.

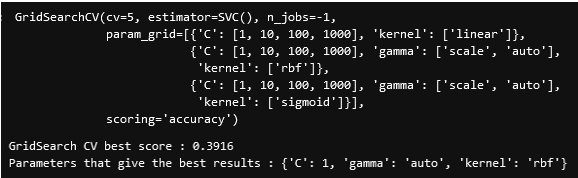
**4.3 Support Vector Machine** When looking into the SVM model, we can use all the variables/features stated above except for the exception of the labels for the personalities. The only difference is in this model we are scaling the variable/features. This allows our machine learning model to tell the difference in weight for each of the variables, which means that *AvSentLength* of 13.6 does not have more weight than the word *good* with the value of just 1. In this model we are using the Standard Scaler. Which assumes the data is normally distributed and then scales each feature around 0 with the standard deviation of 1.



*Figure 12 Data scaled*

Once the data is scaled, we are now able to run our first model with the default values. From sklearn SVC, the default kernel is ‘rbf’ which is a Radial Basis Function. This function is based on the distance from one data point to another. The parameters for the model are cost(C) and gamma. Cost controls the penalty of the outliers. If C is high that means, there are going to be less outliers and higher chance of overfitting and vice versa when C is low. When looking at gamma we are looking at the distance of each point in the group. The lower the gamma the more distance is allowed for that point to be grouped together if they are in the same class. This tends to allow underfitting. When gamma is high then the points will have to be closer to one another in order to be considered in the same group and tends to be over-fitting.

When looking at the accuracy of the multiple models we can see that sigmoid and rbf defaults have the highest score. When we adjusted the parameters, our accuracy started to drop off but when we are checking for under and over fitting. We can see that the sigmoid model is slightly overfitting the model. After understanding the basics of running the SVC() model we are able to apply these to the GridSearchCV(). This package is part of sklearn and what it does is that it allows the user to loop through all the hyperparameters the user sets up. As you can see below we are able to run multiple kernels (linear, rbf, and sigmoid) then we can declare what C and gamma values we want the model to run.



*Figure 13 GridSearchCV*

Once the model was completed we can see that our best score was 0.3916 and our best model out of the input parameters had C =1, gamma set to auto using the ‘rbf’ kernel. Now this is great for us to run one model that will give us the result of which one parameter is the best performance, but the trade off of this model was the time it took to run. Even when we set our n\_jobs to use all cores from our CPU it took over 24 hours to figure out which parameters were the best.

With the data that was given and used for this project, we knew that it was skewed and have tried over and under-sampling to give us a balanced dataset. Yet even balancing out the different types the accuracy shift was between +- 5% which was still a biased dataset. Our approach to this was then breaking the 16 personality types into the 8 characteristics of the personality.

| **Models**  **SVM(rbf)** | **Training**  **Accuracy** | **Valid**  **Accuracy** | **Null**  **Accuracy** | **Precision** | **Recall** |
| --- | --- | --- | --- | --- | --- |
| **I or E** | 0.9072 | 0.7654 | 0.7533 | I(0.77) / E(0.69) | I(0.99) / E(0.09) |
| **N or S** | 0.8729 | 0.8697 | 0.8697 | N(0.87) / S(0) **BIAS** | N(1) / S(0) **BIAS** |
| **P or J** | 0.9144 | 0.6928 | 0.6098 | P(0.71) / J(0.65) | P(0.84) / J(0.46) |
| **F or T** | 0.9323 | 0.7804 | 0.5383 | F(0.79) / T(0.77) | F(0.81) / T(0.74) |

*Table 4 SVM(rbf)*

When comparing the two tables, table 4 and table 6 in appendix , out of the different types of kernel tricks that could have been applied, rbf and sigmoid performed the best compared to poly and linear. Starting from the rbf model we can see that most of the models are under fitting but the null accuracy isn't far off from the validation accuracy, which means that our classifier is predicting the labels correctly. Looking into the precision and recall we can see that I/E, P/J, and F/T are close to each other and are pretty high. Since we are predicting and getting accuracy of each personality we will be looking closer to precision because it focuses more on the correctness of the model. Lastly if we are looking at a sigmoid model we can see that validation accuracy is slightly higher than our training accuracy, which tells us that our model is overfitting. Hence, this led us to conclude with the SVM classifier that we would want to use the rbf model in this case.

**5. Conclusion** From the figure below we can see the accuracy of each of the models we ran. We can also see that NN and SVM are the two models recommended to be used when it comes to predicting individual personality types. Even breaking down into the 8 individual types we can see that our models are scoring higher than the 50% percentile. Overall, to have accuracy this high gives us a clear understanding that these models could help the user to predict either individual types.

| **Validation Accuracy** | **NN** | **Decision Tree** | **SVM** | **Best Model** |
| --- | --- | --- | --- | --- |
| **I/E** | 0.7787 | 0.6484 | 0.7677 | NN/SVM |
| **N/S** | 0.8490 | 0.7504 | 0.8692 | NN |
| **P/J** | 0.6732 | 0.5077 | 0.6893 | NN/SVM |
| **F/T** | 0.7533 | 0.5342 | 0.7620 | NN/SVM |

*Table 5 Accuracy Comparison*

For these models to be deployed in the field, we do strongly recommend them to be at least compared with other personality models to get an understanding of if these models are truly accurate. With Myer's personally it can be used multiple ways from our understanding of them. Examples could range from self-growth, relationship growth (personal and work life), and even dating. In conclusion, by using a person’s biography and applying it to our models we are able to get a “rough” understanding of the person before even knowing them, which can help the user in many ways of approaching the person or even reduce loads from a filtering process.   
 We believe that this project can be useful for a few reasons. This model could be an added part of the test where requests are made for an individual to provide their twitter or facebook posts to get a better match and reduce people rigging it. Recently on dating apps users have been placing their personality type in their bios. This algorithm could take each dater's bio, answers to profile questions and scrub it against this to add their type on their page.

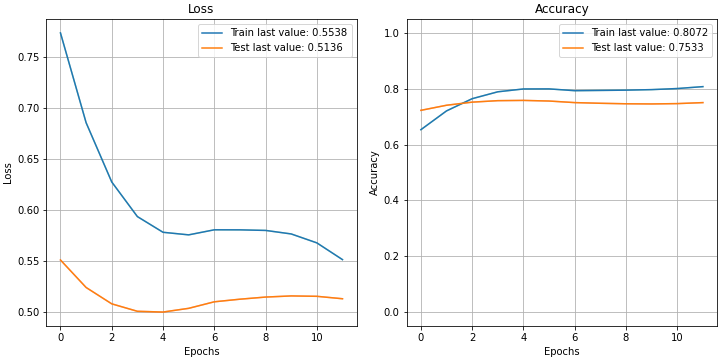
**APPENDIX:**

1. Codes:
   1. Data Cleaning:
      1. Cleaners\_Combined.ipynb
      2. CombinedFeaturesV2.ipynb
      3. FeaturesToOneHots.ipynb
   2. Models:
      1. NeuralNetwork.ipynb
      2. SVM\_Model.ipynb
      3. Decision Tree\_1.py
      4. Decision Tree\_2.py
   3. Exploration:
2. MBTI\_CouldWord\_LexicalDiversity.py

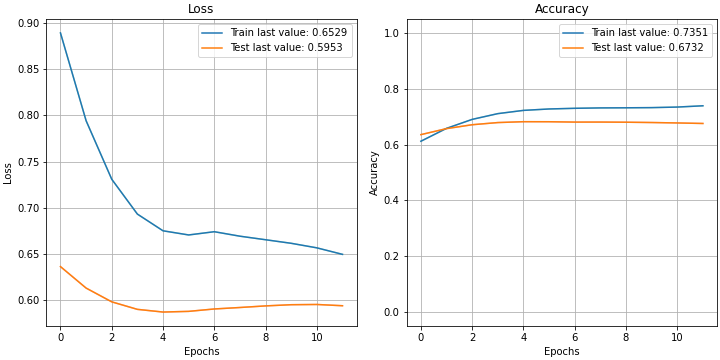
Link to files above: <https://drive.google.com/drive/folders/1y9yGbTOJDdw-3RTCVd853c7iFSgc-qZv?usp=sharing>

1. Neural Network Examples (Figure 14):

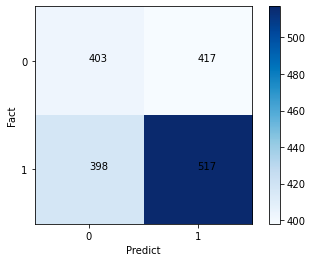
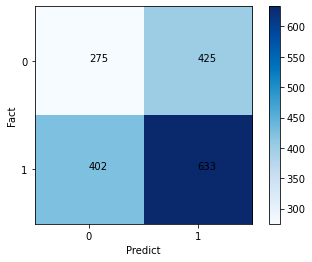
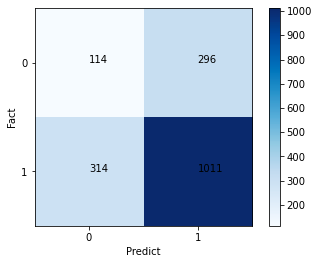
Thinking Versus Feeling (T or F)



Perception Versus Judgement (P or J)



1. Confusion Matrix of Decision Tree



I/E P/J F/T

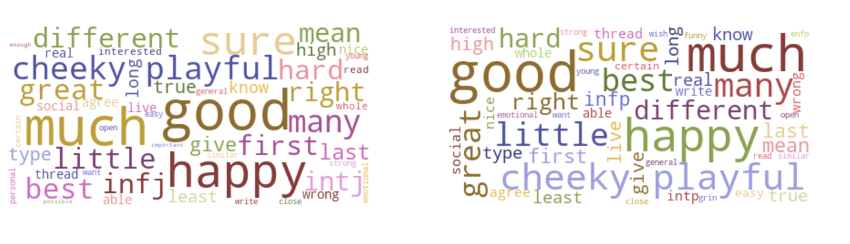
1. Cloud Word



Cloud Word (N)Intuitive and (S)ensing

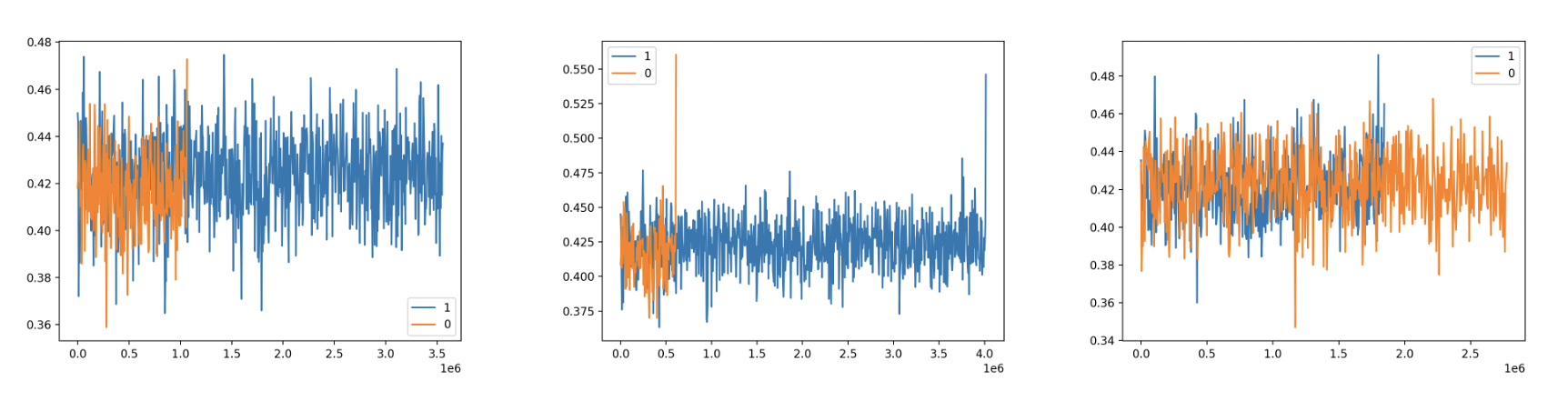


Cloud Word (T)hinking and (F)eeling



Cloud Word (J)udging and (P)erceving

1. Lexical Diversity



(E)xtraversion and (I)troversion (N)Intuitive and (S)ensing (J)udging and (P)ercerving

1. Table 6 - SVM(sigmoid)

| **Models**  **SVM(sigmoid)** | **Training**  **Accuracy** | **Valid**  **Accuracy** | **Null**  **Accuracy** | **Precision** | **Recall** |
| --- | --- | --- | --- | --- | --- |
| **I or E** | 0.7728 | 0.7677 | 0.7533 | I(0.79) / E(0.58) | I(0.95) / E(0.22) |
| **N or S** | 0.8588 | 0.8692 | 0.8697 | N(0.87) / S(0) **BIAS** | N(1) / S(0) **BIAS** |
| **P or J** | 0.6661 | 0.6893 | 0.6098 | P(0.72) / J(0.62) | P(0.79) / J(0.52) |
| **F or T** | 0.7406 | 0.7620 | 0.5383 | F(0.78) / T(0.74) | F(0.78) / T(0.74) |